A Neuro-Evolutionary Approach to Electrocardiographic Signal Classification

Antonia Azzini, Mauro Dragoni, Andrea G. B. Tettamanzi

A Neuro-Evolutionary Approach to Electrocardiographic Signal Classification

Antonia Azzini[1], Mauro Dragoni[2], and Andrea G. B. Tettamanzi[3]

[1] Università degli Studi di Milano
Dipartimento di Tecnologie dell'Informazione
via Bramante, 65 - 26013 Crema (CR) Italy
{antonia.azzini,andrea.tettamanzi}@unimi.it
[2] Fondazione Bruno Kessler (FBK-IRST)
Via Sommarive 18, Povo (Trento), Italy
dragoni@fbk.eu
[3] I3S Laboratory UMR 7271 - UNS/CNRS/INRIA,
Pôle GLC, WIMMICS Research Team, 930 route des Colles – Bât. Les Templiers,
06903, Sophia Antipolis CEDEX, Nice, France
andrea.tettamanzu@unice.fr

Abstract. This chapter presents an evolutionary ANN classifier system as a heartbeat classification algorithm designed according to the rules of the PhysioNet/Computing in Cardiology Challenge 2011 [7], whose aim is to develop an efficient algorithm able to run within a mobile phone, that can provide useful feedback in the process of acquiring a diagnostically useful 12-lead Electrocardiography (ECG) recording.
The method used to solve this problem is to apply a very powerful natural computing analysis tool, namely evolutionary neural networks, based on the joint evolution of the topology and the connection weights relying on a novel similarity-based crossover.
The chapter focuses on discerning between usable and unusable electrocardiograms tele-medically acquired from mobile embedded devices. A preprocessing algorithm based on the Discrete Fourier Trasform has been applied before the evolutionary approach in order to extract the ECG feature dataset in the frequency domain. Finally, a series of tests has been carried out in order to evaluate the performance and the accuracy of the classifier system for such a challenge.

Keywords: Signal Processing, Heartbeat Classification, Evolutionary Algorithms, Neural Networks

1 Introduction

In the last decades, cardiovascular diseases have represented one of the most important causes of death in the world [8] and the necessity of a trustworthy heart state evaluation is increasing. Electrocardiography (ECG) is one of the most useful and well-known methods for heart state evaluation. Indeed, ECG

analysis is still one of the most common and robust solutions in the heart diseases diagnostic domain, also due to the fact that it is one of the simplest non-invasive diagnostic methods for various heart diseases [10].

In such a research field, one of the most important critical aspects regards the quality of such heart state evaluations, since, often, the lack of medically trained experts, working from the acquisition process to the discernment between usable and unusable medical information, increases the need of easy and efficient measuring devices, which can send measured data to a specialist. Furthermore, the volume of the data that have to be recorded is huge and, very often, the ECG records are non-stationary signals, and critical information may occur at random in the time scale. In this situation, the disease symptoms may not come across all the time, but would show up at certain irregular intervals during the day.

In this sense, the Physionet Challenge [7], on which this chapter focuses, aims at reducing, if not eliminating, all the fallacies that currently plague usable medical information tele-medically provided, by obtaining efficient measuring systems through smart phones.

In this challenge, which aim is the ECGs classification, several approaches were explored; in particular, in order to inform inexperienced user about the quality of measured ECGs, artificial-intelligence-based (AI-based) systems have been considered, to reduce the percentage of bad-quality ECGs sent to the specialist and, thus, contribute to a more effective use of her time.

Moody and colleagues [5] reported that some of the top competitors in this challenge employed a variety of techniques, using a wide range of features including entropy, higher order moments, intra-lead information, etc, while the classification methods also included Decision Trees, Support Vector Machines (SVMs), Fuzzy Logic, and heuristic rules.

An example of SVM-based approach is reported in [8], where the authors developed a decision support system based on an algorithm that combines simple rules in order to discard recordings of obvious low quality and a more sophisticated classification technique for improving the quality of AI-based systems for mobile phones, including the fine tuning of detection sensitivity and specificity. Another example has been also given in [9], where a rule-based classification method that mimics the SVM has been implemented, by using a modified version of a real-time QRS-Complex detection algorithm and a T-Wave detection approach.

According to [5], Artificial Neural Networks (ANNs) have been extensively employed in computer aided diagnosis because of their remarkable qualities: capacity of adapting to various problems, training from examples, and generalization capabilities with reduced noise effects. Also Jiang and colleague confirmed the usefulness of ANNs as heartbeat classifiers, emphasizing in particular evolvable ANNs, due to their ability to change the network structure and internal configurations as well as the parameters to cope with dynamic operating environments. In particular, the authors developed an evolutionary approach for

structure and weight optimization of block-based neural network (BbNN) models [18] for a personalized ECG heartbeat pattern classification.

We approach the heartbeat classification problem with another evolutionary algorithm for the joint structure and weights optimization of ANNs [4], which exploits an improved version of a novel similarity-based crossover operator [1], based on the conjunction of topology and connection weight optimization.

This chapter is organized as follows: Section 2 briefly presents the problem, while a description of the evolutionary approach considered in this work is reported in Section 4. The results obtained in the experiments carried out are presented in Section 5, along with a discussion of the performances obtained. Finally, Section 6 provides some concluding remarks.

2 Problem Description

As previously reported, the ECG is a bio-electric signal that records the electrical activities of the heart. It provides helpful information about the functional aspects of the heart and of the cardiovascular system, and the state of cardiac health is generally reflected in the shape of ECG waveform, that is a critical information. For this reason, computer-based analysis and classification and automatic interpretation of the ECG signals can be very helpful to assure a continuous surveillance of the patients and to prepare the work of the cardiologist in the analysis of long recordings.

Moreover, as indicated by the main documentation of Physionet, according to the World Health Organization, cardiovascular diseases (CVD) are the number one cause of death worldwide. Of these deaths, 82% take place in low- and middle-income countries. Given their computing power and pervasiveness, the most important question is to check the possibility, for mobile phones, to aid in delivery of quality health care, particularly to rural populations distant from physicians with the expertise needed to diagnose CVD.

Advances in mobile phone technology have resulted in global availability of portable computing devices capable of performing many of the functions traditionally requiring desktop or larger computers. In addition to their technological features, mobile phones have a large cultural impact. They are user-friendly and are among the most efficient and most widely used means of communication. With the recent progress of mobile-platforms, and the increasing number of mobile phones, a solution to the problem can be the recording of ECGs by untrained professionals, and the subsequent transmission to a human specialist.

The aim of the PhysioNet/Computing in Cardiology Challenge 2011 [6] is to develop an efficient algorithm able to run in near real-time within a mobile phone, that can provide useful feedback to a layperson in the process of acquiring a diagnostically useful ECG recording. In addition to the approaches already cited in Section 1, referring to such a challenge, Table 3 reports other solutions already presented in the literature, capable of quantifying the quality of the ECG looking at individual or combined leads, which can be implemented on

a mobile-platform. As reported later, all such approaches are used to compare their results with those obtained in this work.

3 Neuro-Evolutionary Classifiers

Generally speaking, a supervised ANN is composed of simple computing units (the *neurons*) which are connected to form a network [20–22]. Whether a neuron a influences another neuron b or not depends on the ANN structure. The extent of such influence, when there is one, depends on the weight assigned to each connection among the neurons. It is very difficult to find an optimal network (structure and weights) for a given problem.

Even though some authors do not consider supervised classification as a good domain for neuroevolution, preferring alternatives as support vector machines, Bayesian methods, or analytic optimization methods, neural networks are nevertheless one of the most popular tools for classification. The recent vast research activities in neural classification establish that neural networks are an effective alternative to various conventional classification methods and a large number of successful applications presented in the recent literature demonstrate that ANN design can be further improved by synergetically combining it with evolutionary algorithms, being able to take into account all aspects of ANN design at one time [23].

The review by Zhang [24], which provides a summary of the most important advances in classification with ANNs, makes it clear that the advantages of neural networks lie in different aspects: their capability to adapt themselves to the data without any explicit specification of functional or distributional form for the underlying model; they are universal functional approximators; they represent nonlinear and flexible solutions for modeling real world complex relationships; and, finally, they are able to provide a basis for establishing classification rules and performing statistical analysis. On the other hand, different neuro-evolutionary approaches have been successfully applied to a variety of benchmark problems and real-world classification tasks [25–27, 32]. Our neuro-evolutionary algorithm, too, has been already tested and applied with success to several real-world problems, showing how such an approach can be useful in different classification problems, like automated trading strategy optimization [3, 28], incipient fault diagnosis in electrical drives [29], automated diagnosis of skin diseases [30], etc. Further insights on the evolutionary optimization of ANNs can be found in some broad surveys on the topic [31, 33, 34].

4 The Neuro-Evolutionary Algorithm

The overall algorithm is based on the evolution of a population of individuals, represented by Multilayer Perceptron neural networks (MLPs), through a joint optimization of their structures and weights, here briefly summarized; a more complete and detailed description can be found in [4]. In this work the algorithm uses the Scaled Conjugate Gradient method (SCG) [17] instead of the

more traditional error back-propagation (BP) algorithm in order to speed up the convergence of such a conventional training algorithm. Accordingly, it is the genotype which undergoes the genetic operators and reproduces itself, whereas the phenotype is used *only* for calculating the genotype's fitness. The rationale for this choice is that the alternative of applying SCG to the genotype as a kind of 'intelligent' mutation operator, would boost exploitation while impairing exploration, thus making the algorithm too prone to being trapped in local optima.

The population is initialized with different number of layers and neurons for each individual according to two exponential distributions, in order to maintain diversity among all of them in the new population. Such dimensions are not bounded in advance, even though the fitness function may penalize large networks. The number of neurons in each hidden layer is constrained to be greater than or equal to the number of network outputs, in order to avoid hourglass structures, whose performance tends to be poor. Indeed, a layer with fewer neurons than the outputs destroys information which later cannot be recovered.

Thanks to this encoding, individual ANNs are not constrained to a preestablished topology. Unlike NEAT [19], which starts with minimal network topologies and then applies evolutionary mechanisms to augment them, our approach randomly initializes the network's population with different hidden layer sizes and different numbers of neurons for each individual according to two exponential distributions, in order not to constrain search and to provide a balanced mix of topologies.

4.1 Evolutionary Process

In the evolutionary process, the genetic operators are applied to each network until the termination condition is satisfied, i.e. until the maximum number of generations is reached or no further improvement of the fitness function can be obtained. In each new generation, a new population of size n has to be created, and the first half of such a new population corresponds to the best parents that have been selected by the truncation operator, while the second part of the new population is filled with offspring of the previously selected parents. A child individual is generated by applying the crossover operator to two individuals, selected from the best half of the population (parents), if their local similarity condition is satisfied. Otherwise, the child corresponds to a randomly chosen copy of either of the two selected parents.

Elitism allows the best individual to survive unchanged into the next generation. Then, the algorithm mutates the weights and the topology of the offspring, trains the resulting networks, calculates the fitness on the test set, and finally saves the best individual and statistics about the entire evolutionary process. Although the joint application of truncation selection and elitism could seem to exert a strong selection pressure, which could produce too fast a convergence of the solutions, all the experiments carried out with the `SimBa` [1] crossover outperform the other approaches without showing any premature convergence of the results.

The general framework of the evolutionary process can be described by the following pseudo-code. Individuals in a population compete and communicate with each other through genetic operators applied with independent probabilities, until the termination condition is met.

1. Initialize the population by generating new random individuals.
2. For each genotype, create the corresponding MLP, and calculate its cost and its fitness values.
3. Save the best individual as the best-so-far individual.
4. While not termination condition do:
   (a) Apply the genetic operators to each network.
   (b) Decode each new genotype into the corresponding network.
   (c) Compute the fitness value for each network.
   (d) Save statistics.

The application of the genetic operators to each network is described by the following pseudo-code:

1. Select $\lfloor n/2 \rfloor$ individuals from the population (of size n) by truncation and create a new population of size n with copies of the selected individuals.
2. For all individuals in the population:
   (a) Randomly choose two individuals as possible parents.
   (b) If their local similarity is satisfied
      then generate the offspring by applying crossover according to the crossover probability.
      else generate the offspring by randomly choosing either of the two parents.
   (c) Mutate the weights and the topology of the offspring according to the mutation probabilities.
   (d) Train the resulting network using the training set.
   (e) Calculate the fitness f on the test set.
3. Save the individual with lowest f as the best-so-far individual if the f of the previously saved best-so-far individual is higher (worse).
4. Save statistics.

For each generation of the population, all the information of the best individual is saved.

Table 1 lists all the parameters of the algorithm; their values, reported in the third column, have been experimentally found as those that provide the most satisfactory results.

Selection Truncation selection, the selection method implemented in this work, is taken from the breeder genetic algorithm [37], and differs from natural probabilistic selection in that evolution only considers the individuals that best adapt to the environment. Truncation selection is not a novel solution and previous work considered such a selection in order to prevent the population from remaining too static and perhaps not evolving at all [35]. It is a very simple technique which produces satisfactory solutions in conjunction with other strategies, like elitism, which allows the best individual to survive unchanged into the next generation and solutions to monotonically get better over time.

Table 1. Parameters of the Algorithm.

Symbol	Meaning	Default Value
n	Population size	60
p^{+}_{layer}	Probability of inserting a hidden layer	[0.05,0.15,0.30,0.45]
p^{-}_{layer}	Probability of deleting a hidden layer	[0.05,0.15,0.30,0.45]
p^{+}_{neuron}	Probability of inserting a neuron in a hidden layer	[0.05,0.15,0.30,0.45]
p_{cross}	'Desired' probability to apply crossover	[0.2,0.4,0.6,0.8,1.0]
δ	Crossover similarity cut-off value	0.1
N_{in}	Number of network inputs	$*$)
N_{out}	Number of network outputs	$*$)
α	Cost of a neuron	2
β	Cost of a synapse	4
λ	Desired trade-off between network cost and accuracy	0.2
k	Constant for scaling cost and MSE in the same range	10^{-6}

$*$) Benchmark dataset dependent.

Mutation The main function of this operator is to introduce new genetic material and to maintain diversity in the population. Generally, the purpose of mutation is to simulate the effects of transcription errors that can occur with a very low probability, the mutation rate, when a chromosome is replicated. The evolutionary process applies two kinds of neural network perturbations: weights mutation and topology mutation.

Weights mutation perturbs the weights of the neurons before performing any structural mutation and applying SCG. This kind of mutation uses a Gaussian distribution with zero mean and variance given by matrix $\mathbf{Var}^{(i)}$ for each network. as illustrated in Table 2. This solution is similar to the approach implemented by *evolution strategies* [36], algorithms in which the strategy parameters are proposed for self-adapting the mutation concurrently with the evolutionary search. The main idea behind these strategies is to allow a control parameter, like mutation variance, to self-adapt rather than changing its value according to some deterministic algorithm. Evolution strategies perform very well in numerical domains, and are well-suited to (real) function optimization. This kind of mutation offers a simplified method for self-adapting each single value of the Variance matrix $\mathbf{Var}^{(i)}_{j}$, whose values are defined as log-normal perturbations of their parent parameter values.

Topology mutation can apply four types of mutation by considering neurons and layer addition and elimination. The addition and the elimination of a layer and the insertion of a neuron are applied with independent probabilities, corresponding respectively to the three algorithm parameters p^{+}_{layer}, p^{-}_{layer}, and p^{+}_{neuron}, while a neuron is eliminated only when its contribution becomes negligible (less that 5%) with respect to the overall behavior of the network [3]. The parameters used in such kind of mutation are set at the beginning and maintained unchanged during the entire evolutionary process.

All the topology mutation operators are aimed at minimizing their impact on the behavior of the network; in other words, they are designed to be as little disruptive, and as much neutral, as possible, preserving the effectiveness between the parent and the offspring better than by adding random nodes or layers.

Fitness Function Although it is customary in EAs to assume that better individuals have higher fitness, as previously considered [3, 2], the convention that a lower fitness means a better ANN is adopted in this work. This maps directly to the objective function of an error- and cost- minimization problem, which is the natural formulation of most problems ANNs can solve.

The fitness function is calculated, after the training and the evaluation processes, by Equation 1 and it is defined as a function of the confusion matrix M obtained by that individual:

$$f_{multiclass}(M) = N_{\text{outputs}} - \text{Trace}(M), \tag{1}$$

where N_{outputs} is the number of output neurons and $\text{Trace}(M)$ is the sum of the diagonal elements of the row-wise normalized confusion matrix, which represent the conditional probabilities of the predicted outputs given the actual ones.

Crossover in general, recombination in EAs is most useful before convergence, during the exploration phase, when it may help locating promising areas of the search space. This is exactly why the local-similarity based crossover presented in this work turns out to be beneficial. Indeed, this operator exploits the concept of similarity among individuals, which is one of the first ideas developed in the literature for solving such problem.

We have given particular attention to two empirical studies [38, 39], which focus on the evolution of the single network unit involved in the crossover operator, the hidden node. Their aim was to emphasize the equivalence between hidden nodes of ANNs, in order to identify similarly performing units prior to crossover, avoiding all the disruptive effects stated above. Following such an idea, we extend our neuro-genetic approach already presented in the literature [3, 4], which implements a joint optimization of weights and network structure, by defining a novel crossover operator. This operator allows recombination of individuals that have different topologies, but with hidden nodes that are similarly performing in the cutting point of the hidden layer randomly chosen (indicated in the approach as *local similarity*). The evolutionary process does not consider only a part, but complete multilayer perceptrons (MLPs), achieving satisfactory performances and generalization capabilities, as well as reduced computational costs and network sizes.

The crossover is applied with a probability parameter p_{cross}, defined by the user together with all the other genetic parameters, and maintained unchanged during the entire evolutionary process. A significant aspect related to the crossover probability considered in this work is that it refers to a 'desired'

probability, a genetic parameter set at the beginning of the evolutionary process indicating the probability with which the crossover should be applied. However, the 'actual' crossover probability during the execution of the algorithm is less than or equal to the desired one, because the application of the crossover operator is conditional on a sort of 'compatibility' of the individuals involved. We report a summary of the crossover operator, already discussed in the literature [1, 2].

The `SimBa` crossover starts by looking for a 'local similarity' between two individuals selected from the population. We refer to 'local similarity' as a situation in which, in both individuals, there are two consecutive layers (i and $i + 1$) with the same number of neurons. This is a necessary condition for the application of our crossover operator because its aim is to overcome the problem related with structure incompatibility between individuals. The contribution of each neuron of the layers selected for the crossover is computed, and the neurons of each layer are reordered according to their contribution (i.e., the output obtained by evaluating the neural network, up to that neuron, over the training dataset), which strongly depends on its input connections. We also choose a cut-off threshold δ that will be used for swapping the neurons.

Then, each neuron of the layer selected in the first individual is associated with the most 'similar' neuron (a neuron with the most similar output) in the other's individual layer, and the neurons of the layer of the second individual are re-ranked by considering the associations with the neurons of the first one. All the neuron associations linked with an output difference higher than the cut-off value δ are discarded. Such cut-off value δ can be seen as the 'local-similarity' threshold. In this work it has been experimentally defined at the beginning equal to 0.1 that is maintained unchanged during the entire evolutionary process. Only the neurons above such similarity threshold are eligible for being swapped, while the others will remain unchanged. Finally a cut-point is randomly selected and the neurons above the cut-point are swapped by generating the offspring of the selected individuals.

5 Experiments and Results

The data used for the PhysioNet/CINC 2011 Challenge consist of 2,000 twelve-lead ECGs (I, II, III, aVR, aVF, aVL, V1, V2, V3, V4, V5, and V6), each 10 second long, with a standard diagnostic bandwidth defined in the range (0.05–100 Hz). The twelve leads are simultaneously recorded for a minimum of 10 seconds; each lead is sampled at 500 Hz with 16-bit resolution.

The proposed approach has been evaluated by using the dataset provided by the challenge organizers. This dataset, described above in Section 2, is public and has been distributed in two different parts:

- Set A: this dataset has to be used to train the approach. It is composed of 998 instances provided with reference quality assessments;
- Set B: this dataset has to be used for testing the approach. It is composed of 500 instances and the reference quality assessments are not distributed to

the participants. The reports generated by the approach must be sent to the submission system in order to receive results valid for the challenge.

We split Set A in two parts: a training set consisting of 75% of the instances contained in Set A, and a validation set, used to stop the training algorithm, consisting of the remaining 25%, while Set B is used as test set for the final evaluation of the approach.

Each instance of the dataset represents an ECG signal composed of 12 series (one for each lead) of 5,000 values representing the number of recordings performed for each lead. These data have been preprocessed in order to extract the features that we used to create the datasets given in input to the algorithm. We have applied the FFT to each lead in order to transform each lead to the frequency domain. After the transformation, we summed the 5,000 values by groups of 500 in order to obtain 10 features for each lead. Finally, The input attributes of all datasets have been rescaled, before being fed as inputs to the population of ANNs, through a Gaussian distribution with zero mean and standard deviation equal to 1.

The experiments have been carried out by setting the parameters of the algorithm to the values obtained in a first round of experiments aimed at identifying the best parameter setting. These parameter values are reported in Table 2. We performed 40 runs, with 40 generations and 60 individuals for each run, while the number of epochs used to train the neural network implemented in each individual has been set to 250.

Table 2. Parameters of the Algorithm.

Symbol	Meaning	Default Value
n	Population size	60
p^{+}_{layer}	Probability of inserting a hidden layer	0.05
p^{-}_{layer}	Probability of deleting a hidden layer	0.05
p^{+}_{neuron}	Probability of inserting a neuron in a hidden layer	0.05
p_{cross}	'Desired' probability of applying crossover	0.7
δ	Crossover similarity cut-off value	0.9
N_{in}	Number of network inputs	120
N_{out}	Number of network outputs	1
α	Cost of a neuron	2
β	Cost of a synapsis	4
λ	Desired tradeoff between network cost and accuracy	0.2
k	Constant for scaling cost and MSE in the same range	10^{-6}

The challenge has been organized in two different events: a closed event and an open one. While in the closed event it is possible to develop the classification algorithm in any language, in the open event it is mandatory to develop the algorithm in Java. For this reason, considering that the proposed approach has been

developed in Java too, we compared the results we obtained to those obtained by the other systems that participated in the challenge in the open event. It is important to highlight that we do not claim to obtain the best performance, but our goal was to show that, even if our system is trained with a training set that exploits very little information, the performance obtained by our approach does not lag too much behind the one obtained by the best state-of-the-art systems.

Table 3 shows the results obtained by the other participants compared with the results obtained by the proposed approach. Besides comparing our approach with the other approaches presented at the challenge, we have also compared it with the other following neuro-genetic approaches:

- Simple ANN with Conjugated Gradient: the classifiers are encoded with a population of ANNs, trained with the Conjugated Gradient method over 1,000,000 epochs. Also in this case, the networks are then evaluated over the validation and the test sets, respectively, through the application of the mean square error.
- NeuroEvolution of Augmenting Topologies (NEAT) approach [19]: an evolutionary approach applied to neural network design that: (1) uses a crossover on different topologies, (2) protects structural innovation by using speciation, and (3) applies an incrementally growing from minimal network structures.
- Evolved ANN without crossover: the population of ANNs are evolved through the joint optimization of architecture and connection weights reported in this chapter, but in this case no crossover is implemented. The number of epochs corresponds to 250.

We report both the best and the average performance obtained by the proposed approach. It is possible to observe that, if we consider the best performance, we obtained the second best accuracy; while the average accuracy, computed over the 40 runs, obtained the fourth performance. The robustness of the approach is also proved by observing the low value of the standard deviation that, in the performed experiments, was 0.011. In italics, we show the performance obtained by the other approaches that we have used for classifying the data in order to compare them with the approach proposed in this paper. The results demonstrated that the proposed approach outperforms the other ones. Indeed, the NEAT approach obtained only the seventh accuracy, while the other two approaches obtained respectively the eighth and the eleventh performance.

Besides the evaluation on the test set, we performed also a ten-fold cross validation on the training set. We split the training set in ten folds F_i and we performed ten different set of 10 runs in order to observe which is the behavior of the algorithm when training, validation, and test data change. Table 4 shows the results of the ten-fold cross validation. By observing the results we can observe the robustness of the algorithm. In fact, the accuracies obtained by changing the folds used for training, validation, and test are very close; moreover, the standard deviation of the results is very low.

Table 3. Results of the open event challenge.

Participant	**Score**
Xiaopeng Zhao [11]	0.914
Proposed Approach (Best)	**0.902**
Benjamin Moody [12]	0.896
Proposed Approach (Average)	**0.892**
Lars Johannesen [13]	0.880
Philip Langley [14]	0.868
NEAT (Average)	*0.856*
Evolved ANN without crossover (Average)	*0.845*
Dieter Hayn [15]	0.834
Vclav Chudcek [16]	0.833
Simple ANN with Conjugated Gradient (Average)	*0.818*

Table 4. Results of the ten-fold cross validation.

Training Set	**Validation Set**	**Test Set**	**Avg Accuracy**	**Std Deviation**
F1...F7	F8, F9	F10	0.8984	0.0035
F2...F8	F9, F10	F1	0.8988	0.0067
F3...F9	F10, F1	F2	0.9002	0.0075
F4...F10	F1, F2	F3	0.9022	0.0107
F5...F10, F1	F2, F3	F4	0.9040	0.0071
F6...F10, F1, F2	F3, F4	F5	0.9002	0.0029
F7...F10, F1...F3	F4, F5	F6	0.9002	0.0018
F8...F10, F1...F4	F5, F6	F7	0.8976	0.0054
F9, F10, F1...F5	F6, F7	F8	0.9032	0.0090
F10, F1...F6	F7, F8	F9	0.8986	0.0047

6 Conclusions

In this chapter, we have proposed an ECG classification scheme based on a neuro-evolutionary approach, based on the joint evolution of the topology and the connection weights together with a novel similarity-based crossover, to aid classification of ECG recordings. The signal were first transformed into the frequency domain by using a Fast Fourier Trasform algorithm, and then they were normalized through a gaussian distribution with 0 mean and standard deviation equal to 1. The present system was validated on real ECG records taken from the PhysioNet/Computing in Cardiology Challenge 2011.

A series of tests has been carried out in order to evaluate the capability of the neuro-evolutionary approach to discern between usable and unusable electrocardiograms tele-medically acquired from mobile embedded devices. The results show an overall satisfactory accuracy and performances in comparison with other approaches carried out in this challenge and presented in the literature.

It is important to stress the fact that the proposed method was able to achieve top-ranking classification accuracy despite the use of a quite standard preprocessing step and a very small number of input features. No attempt was made yet to fine tune the signal pre-processing and the feature selection steps. On the other hand, it is well known that these two steps are often critical for the success of a signal classification methods. For this reason, we believe that the proposed neuro-evolutionary approach has a tremendous improvement potential.

Future work will involve the adoption of more sophisticated preprocessing techniques, by working, for example, on a multi-scale basis, where each scale represents a particular feature of the signal under study. Other ideas could regard the study and the implementation of feature selection algorithms in order to provide an optimized selection of the signal given as inputs to the neural networks.

References

1. A. Azzini, A.G.B. Tettamanzi, M. Dragoni: SimBa-2: Improving a Novel Similarity-Based Crossover for the Evolution of Artificial Neural Networks. In: 11th International Conference on Intelligent Systems Design and Applications (ISDA 2011), pp. 374–379. IEEE (2011)
2. A. Azzini, M. Dragoni, A.G.B. Tettamanzi: A novel similarity-based crossover for artificial neural network evolution. In: Parallel Problem Solving from Nature PPSN XI, Lecture Notes in Computer Science. vol. 6238, pp. 344–353. Springer (2010)
3. A. Azzini, A.G.B. Tettamanzi: Evolving neural networks for static single-position automated trading. Journal of Artificial Evolution and Applications 2008(Article ID 184286), 1–17 (2008)
4. A. Azzini, A.G.B. Tettamanzi: A new genetic approach for neural network design. In: Engineering Evolutionary Intelligent Systems. Studies in Computational Intelligence. vol. 82. Springer (2008)
5. K. Silva, G.B. Moody, L. Celi: Improving the Quality of ECGs Collected Using Mobile Phones: The PhysioNet/Computing in Cardiology Challenge 2011. Contribution sent to the 38th Physionet Cardiology Challenge (2011).

6. PhysioNet: Research Resource for Complex Physiologic Signals, `http://www.physionet.org`
7. G.B. Moody: Improving the quality of ECGs collected using mobile phones: The 12th Annual Physionet/Computing in Cardiology Challenge. Computing in Cardiology Challenge 38 (2011).
8. Data Driven Approach to ECG Signal Quality Assessment using Multistep SVM Classification. Contribution sent to the 38th Physionet Cardiology Challenge (2011).
9. T.H.C. Tat, C. Chen Xiang, L.E. Thiam: Physionet Challenge 2011: Improving the Quality of Electrocardiography Data Collected Using Real Time QRS-Complex and T-Wave Detection. Contribution sent to the 38th Physionet Cardiology Challenge (2011).
10. S. Jokic, S. Krco, V. Delic, D. Sakac, Jokic, I., Lukic, Z.: An Efficient ECG Modeling for Heartbeat Classification. IEEE 10th Symposium on Neural Network Applications on Electrical Engineering, NEUREL 2010, Belgrade, Serbia, September, 23-25, 2010.
11. H. Xia, J. McBride, A. Sullivan, T. De Bock, J. Bains, D. Wortham, X. Zhao: A Multistage Computer Test Algorithm for Improving the Quality of ECGs. Contribution sent to the 38th Physionet Cardiology Challenge (2011).
12. B.E. Moody: A Rule-Based Method for ECG Quality Control. Contribution sent to the 38th Physionet Cardiology Challenge (2011).
13. L. Johannesen: Assessment of ECG Quality on an Android Platform. Contribution sent to the 38th Physionet Cardiology Challenge (2011).
14. P. Langley, L. Di Marco, S. King, C. Di Maria, W. Duan, M. Bojarnejad, K. Wang, D. Zheng, J. Allen, A. Murray: An Algorithm for Assessment of ECG Quality Acquired Via Mobile Telephone. Contribution sent to the 38th Physionet Cardiology Challenge (2011).
15. D. Hayn, B. Jammerbund, G. Schreier: Real-time Visualization of Signal Quality during Mobile ECG Recording. Contribution sent to the 38th Physionet Cardiology Challenge (2011).
16. V. Chudcek, L. Zach, J. Kulek, J. Spilka, L. Lhotsk: Simple Scoring System for ECG Signal Quality Assessment on Android Platform. Contribution sent to the 38th Physionet Cardiology Challenge (2011).
17. Hestenes, M., Stiefel, E.: Methods of conjugate gradients for solving linear systems. Journal of Research of the National Bureau of Standards 49(6) (1952).
18. W. Jiang, S.G. Kong. Block-Based Neural Networks for Personalized ECG Signal Classification. IEEE Transactions on Neural Networks, 18(6) (2007).
19. K. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. Evolutionary Computation, 10:99–127, 2002.
20. F. Camargo. Learning algorithms in neural networks. Technical Report, Computer Science Department, Columbia University, 1990.
21. K. Gurney. An Introduction to Neural Networks. Taylor & Francis, Inc., Bristol, PA, USA, 1997.
22. B. Kröse, P. Van der Smagt. An introduction to neural networks. URL ftp://ftp.informatik.uni-freiburg.de/papers/neuro/ann_intro_smag.ps.gz, The University of Amsterdam, 1996.
23. X. Yao. Evolving artificial neural networks. Proceedings of the IEEE, pp. 1423–1447, 1999.
24. G. Zhang. Neural networks for classification: A survey. IEEE Transaction on Systems, Man, and Cybernetics - Part C: Applications and Reviews, vol. 30, pp. 451–462, 2000.

25. A. Bahrammirzaee, A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems. Neural Computing and Applications, vol. 19, pp. 1165–1195, 2010.
26. M. Castellani, H. Rowlands. Evolutionary artificial neural network design and training for wood veneer classification. Engineering Applications of Artificial Intelligence, vol. 22, pp. 1165–1195, 2009.
27. J. Fernández, C. Hervás, F. Martínez-Estudillo, P. Gutiérrez. Memetic pareto evolutionary artificial neural networks to determine growth/no-growth in predictive microbiology. Applied Soft computing, vol. 11, pp. 534–550, 2011.
28. A. Azzini, C. da Costa Pereira, A.G.B. Tettamanzi. Modeling turning points in financial markets with soft computing techniques. In: A. Brabazon, M. ONeill, D. Maringer (Eds.). Natural Computing in Computational Finance, vol. 293 of Studies in Computational Intelligence, pp. 147–167, Springer Berlin / Heidelberg,2010.
29. A. Azzini, M. Lazzaroni, A.G.B. Tettamanzi. Incipient fault diagnosis in electrical drives by tuned neural networks. In: Instrumentation and Measurement Technology Conference, IMTC'06, pp. 1284–1289, 2006.
30. A. Azzini, S. Marrara. Dermatology disease classification via novel evolutionary artificial neural network. Proceedings of the 18th International Conference on Database and Expert Systems Applications, DEXA'07, pp. 148–152, 2007.
31. A. Azzini, A.G.B. Tettamanzi. Evolutionary ANNs: A state of the art survey. Intelligenza Artificiale, vol. 5, pp. 19–35, 2011.
32. P. Wang,T. Weise, R. Chiong. Novel evolutionary algorithms for supervised classification problems: an experimental study. Evolutionary Intelligence, vol.: Special Issue, pp. 1–14, 2011.
33. X. Yao, Y. Xu. Recent advances in evolutionary computation. International Journal on Computer Science and Technology, vol. 21, pp. 1–18, 2006.
34. D. Floreano, P. Durr, C. Mattiussi. Neuroevolution: from architectures to learning. Evolutionary Intelligence, vol. 10, pp. 47–62, 2008.
35. D. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Kluwer Academic Publishers, Boston, MA, 1989.
36. H. Schwefel. Numerical Optimization for Computer Models. John Wiley, Chichester, UK, 1981.
37. H. Muhlenbein, D. Schlierkamp-Voosen. The science of breeding and its application to the breeder genetic algorithm (bga). Evolutionary Computation, vol. 1, pp. 335–360, 1993.
38. N. Garcia-Pedrajas, D. Ortiz-Boyer and C. Hervas-Martinez. An alternative approach for neural network evolution with a genetic algorithm: Crossover by combinatorial optimization. Neural Networks, vol. 19, pp. 514–528, 2006.
39. P.J.B. Hancock. Genetic Algorithms and permutation problems: a comparison of recombination operators for neural net structure specification. Proc. of IEEE Int. Workshop on Combinations of Genetic Algorithms and Neural Networks, COGANN'92, pp. 108–122, Baltimore, MD, IEEE Press, 1992.